*Protocolo de ligação lógica*

O objetivo do deste protocolo é fornecer um serviço de comunicação de dados fiável entre dois sistemas ligados por um meio (canal) de transmissão – neste caso, um cabo de série. As principais funções deste protocolo são:

* Configurar a porta de série e guardar as configurações antigas de modo a serem repostas facilmente.
* Estabelecer e terminar as ligações da porta de série
* Delimitação (*framing*) e numeração de tramas.
* Envio e receção de tramas através da porta de série.
* Fazer *Stuffiing* e *Destuffing* das tramas recebidos da Camada de Aplicação.
* Controlo de erros.

**LLOPEN**

Esta função tem a responsabilidade de estabelecer a ligação entre o emissor e o recetor.

No emissor, esta função envia a trama de controlo *SET* e ativa o temporizador que é desativado depois de receber resposta (*UA*). Se não receber resposta dentro de um tempo *time-out*, *SET* é reenviado. Este mecanismo de retransmissão só é repetido um número máximo de vezes, se este número for atingido o programa termina.

No recetor, esta função espera pela chegada de uma trama de controlo SET para responder com um UA.

Para enviar tanto o *UA* como o *SET* é usada a função ***sendControlMessage***. Esta função recebe como argumento o campo de controlo das tramas de Supervisão, o que permite que a função seja usada para enviar qualquer tipo de trama deste tipo.

Como o mecanismo de receção do *UA* necessita de temporizador, usa-se a função ***stateMachineUA*** que tem o que é necessário para tal. Na leitura do *SET* é usada a função ***readControlMessage*** que recebe como argumento o campo de controlo das tramas de Supervisão para verificar se a trama recebida é de facto do tipo que é desejado.

As escritas são feitas trama a trama, no entanto a leitura é feita carater a carater.

**LLWRITE**

Esta é a função no emissor responsável pelo envio das tramas e pelo *stuffing* das mesmas.

Primeiro é feito o *framing* da mensagem, ou seja, acrescentado o cabeçalho do Protocolo de Ligação à mensagem (para calcular o *BCC2* é chamada a função ***calculoBCC2***). Seguidamente é feito o *stuffing* da mensagem e do *BCC2* (feito na função ***stuffingBCC2***). Posto isto, a trama está pronta a ser enviada. Esta escrita é feita trama a trama.

Antes da escrita são introduzidos erros pelas funções ***messUpBCC1*** e ***messUpBCC2***. Estas funções substituem o conteúdo de uma posição aleatória (no caso do *BCC1* da posição 1 a 3 e no caso do *BCC2* nas posições ocupadas pelo campo de dados e *BCC2*) com uma letra aleatória, tendo em conta uma probabilidade erro escolhida.

O envio da trama tem o mesmo mecanismo de *time-out* e retransmissão que o envio do *SET* no ***llopen***. Ou seja, depois de enviar a trama é acionado um alarme até à receção de uma resposta (*RR* ou *REJ*) e se atingido esse alarme a mensagem é reenvida (mecanismo que se pode ocorrer um numero máximo de vezes). Se recebido um *REJ* a mensagem é reenviada. Para fazer a verificação se é *REJ* ou *RR* é usada a função ***readControlMessage*** que retorna o campo de controlo da trama de Supervisão lida.

**LLREAD**

Esta é a função no recetor responsável pela receção das tramas e pelo *destuffing* das mesmas.

A leitura é feita carater a carater. Para verificar o *BCC2* é usada a função ***checkBCC2***, caso esteja correto é enviado *RR*, caso contrário *REJ* usando a função ***sendControlMessage***. O campo de controlo enviado depende se o número de sequência da trama é 0 ou 1. Seguidamente é feito o *destuffing* do campo de dados. Também é feita uma verificação se o número de sequência de tramas é o esperado para ser possível tratar duplicados.

**LLCLOSE**

Esta função tem a responsabilidade de terminar a ligação entre o emissor e o recetor.

No emissor, é enviado a trama de Supervisão *DISC* com ajuda da função ***sendControlMessage*** (que recebe como argumento o campo de controlo da trama a enviar) e esperado outro *DISC* de volta pela função ***readControlMessage*** (que retorna o campo de controlo da trama lida). Para finalizar é enviado um *UA*.

No recetor é esperado um *DISC*, enviado um *DISC* e esperado um UA com as funções ***readControlMessage*** e ***sendControlMessage*** (que recebem ambas o campo de controlo pretendido).